

AD-A094 888

NAVAL RESEARCH LAB WASHINGTON DC
EVALUATION OF THE A-7 REQUIREMENTS DOCUMENT BY ANALYSIS OF CHAN-ETC(U)
DEC 80 V R BASILI, D M WEISS
NRL-0445

F/O 9/2

NL

UNCLASSIFIED

1 of 1
AD-A
000000



END

DATE

FILED

3-81

DTIC

LEVEL

12
B.2

NRL Report 8445

Evaluation of the A-7 Requirements Document by Analysis of Change Data

VICTOR R. BASILI

*Information Processing Systems Branch
Communications Sciences Division
and
University of Maryland*

DAVID M. WEISS

*Information Processing Systems Branch
Communications Sciences Division*

December 29, 1980

DTIC
ELECTE
FEB 11 1981



**NAVAL RESEARCH LABORATORY
Washington, D.C.**

Approved for public release; distribution unlimited.

81 2 11 042

AD A094838

DDC FILE COPY

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NRL Report-8445	2. GOVT ACCESSION NO. AD-A094888	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) EVALUATION OF THE A-7 REQUIREMENTS DOCUMENT BY ANALYSIS OF CHANGE DATA.		5. TYPE OF REPORT & PERIOD COVERED Interim report on a continuing NRL problem
7. AUTHOR(s) Victor R. Basili and David M. Weiss		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, DC 20375		8. CONTRACT OR GRANT NUMBER
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Naval Research Laboratory Washington, DC 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Element 61153N Project RR014-09-41 NRL Problem 75-0199-0-0
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE December 29, 1980
15. SECURITY CLASS (of this report) Unclassified		13. NUMBER OF PAGES 20
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release, distribution unlimited.		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Software engineering Software methodology evaluation Software change data		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) We describe in this report an effective data collection method for evaluating software development methodologies, from definition of the objectives of the data collection to analysis of the results. We show how the data analysis can answer questions with respect to how successfully the goals of the development methodology are met. The A-7 requirements document is used as an example. We provide the results of data analyses conducted partway through the A-7 flight software development cycle, and we discuss the utility of information obtained by such partial analyses. Results from the study show that data collection is feasible and useful when performed as part of — (Continued)		

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601


1. SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

251950

7.5

20. ABSTRACT (Continued)

configuration control, that data distributions based on partial data provided useful feedback to the developers, and that the A-7 requirements document is easily maintained and changed.



CONTENTS

INTRODUCTION	1
A-7 PROJECT OVERVIEW	1
DATA COLLECTION	2
Goals	2
Identification of Data To Be Collected	2
Forms Design	3
Data Collection Procedures	4
Data Validation and Analysis	4
RESULTS OF THE DATA ANALYSIS	4
Questions with Preliminary Answers	8
Questions for Which Lack of Data May Be Meaningful	14
Questions Not Currently Answerable	15
CONCLUSIONS	15
ACKNOWLEDGMENTS	16
REFERENCES	16

Accession For	
NTIS CR&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Avail and/or	
Special	
Dist	
A	

EVALUATION OF THE A-7 REQUIREMENTS DOCUMENT BY ANALYSIS OF CHANGE DATA

INTRODUCTION

In recent years a number of techniques for improving the reliability and decreasing the cost of software have been suggested. These techniques deal with various aspects of the software life cycle. An integrated set of two or more techniques covering one or more phases of the life cycle may be defined as a methodology. It is not obvious how to refine and adjust the basic techniques in a methodology for the individual factors of some specific environment and application. Software engineering involves the application of a methodology to a particular environment.

The software community is interested in the analysis of techniques, their integration into a methodology, and the engineering of that methodology to particular environments. An effective way to evaluate a methodology, understand the environment, and refine the methodology for the environment is to collect data that characterize the methodology and the environment and supply insight into both.

A major source of insight when analyzing a software development project is a record of the changes, including error corrections, made as the development progresses. Data showing where changes were made, what kinds of changes were made, and the effort involved in making changes can be used to evaluate methodologies, characterize environments, and permit the proper engineering of the methodologies for the environments.

We describe in this report an effective data collection method, from definition of objectives of the data collection to analysis of results. We show how analysis of the data can answer questions with respect to how successfully the goals of the development methodology are met. The A-7 requirements document is used as an example. We provide the results of data analyses conducted partway through the A-7 flight software development cycle, and we discuss the utility of information obtained by such partial analyses. The next section describes the project studied and its overall objectives. The third section discusses the relation between the data collection methodology and the software development methodology. The fourth section presents the data and its analysis. The fifth section presents the conclusions and some suggestions for further studies.

A-7 PROJECT OVERVIEW

A significant obstacle in the field of software engineering is lack of technology transfer. Many techniques are developed in academic environments or in the construction of small programs. Large-scale software developers are reluctant to use techniques that have not been tested in the development of complex systems. The Naval Research Laboratory (NRL) and the Naval Weapons Center (NWC) are redesigning and rebuilding the operational flight program for the A-7 aircraft using techniques such as information hiding [1,2], formal specifications [3-6], abstract interfaces [7], cooperating sequential processes [8,9], process synchronization routines [8], and resource monitors [10-12]. The goals are to demonstrate the feasibility of using these techniques to develop a complex, real-time program and to provide the Navy with a model for the development of avionics programs. The techniques to be used were selected because they are claimed to facilitate the development of software that is reliable, easy to change, easy to understand, and easy to demonstrate correct.

The characteristics of the A-7 operational flight program and the constraints on the redevelopment project are described in Ref. 13. The requirements description was completed in November 1978, and the program is currently in the design stage.

DATA COLLECTION

Goals

The opportunity to apply recent software engineering technology in the development of a complex model system does not seem to occur often. We considered it important not to lose the chance to monitor closely the progress of the project. A separate data collection effort to permit evaluation of the project during the development cycle was established. Final evaluation of the success of the rebuilt A-7 software must await the delivery and use of the software. A number of intermediate evaluation points may be established to provide some insight into the redevelopment process. The intermediate evaluations may be based on the goals established at each phase of the project and on the goals established for the different techniques used. As an example, Heninger has described the following six objectives of the requirements document [13,14]:

- Specify external behavior only.
- Specify constraints on the implementation.
- Be easy to change.
- Serve as a reference tool.
- Record forethought about the life cycle of the system.
- Characterize acceptable responses to undesired events.

The main purpose of the data collection and analysis described here is to help measure the success with which the preceding objectives are met.

Identification of Data to be Collected

Once the decision to monitor the project was made and the objectives for the document were clearly stated, the next step was to identify the data to be collected. To do this we established a list of questions, the answer to each question helping to measure the success of attainment of an objective. As an example, consideration of the objective "Be easy to change" led to the following questions:

- Is the document easy to change?
- Is it clear where a change has to be made?
- Are changes confined to a single section?

To answer these questions we needed to know, for each change, the effort required to make the change, some measure of how much of the document had to be examined to make the change, and how many sections of the document were actually modified when the change was made. We decided to measure effort in man-hours and both the amount of the document examined and the amount modified in making the change in sections of the document. The complete list of questions, taken from Ref. 15, is shown in Table I.

Table 1 — Questions Used in Designing the Change Report Form

- (1) Is externally visible behavior only specified without implying a particular implementation?
- (2) Are the appropriate external interfaces specified?
- (3) Are the external interfaces specified correctly?
- (4) Is the document easy to change?
- (5) Is it clear where a change has to be made?
- (6) Are the changes likely to occur predicted correctly?
- (7) Are changes confined to a single section?
- (8) Is the proper set of undesired events described?
- (9) Is the notation used unambiguous?
- (10) Which sections have the most errors?
- (11) Where do the most changes have to be made?
- (12) Which type of tables has the most errors?
- (13) Does the document contain unnecessary information?
- (14) What use of the document reveals the most errors?
- (15) Are sections 3 (Modes) and 4 (Functions) consistent with each other?
- (16) Is the dictionary complete, correct, and consistent with the rest of the document, and will it remain so?
- (17) Which subsections of sections 2 (Data Items), 3 (Modes), and 4 (Functions) are most error-prone?

Forms Design

Experience gained in designing and using change report forms for NASA's Software Engineering Laboratory [16] and for the Architecture Research Facility study [17] helped considerably in the design of the A-7 change report forms. Among the lessons learned from those projects were the following:

- The form should fit on one piece of paper.
- Providing space on the form for brief written descriptions of changes was helpful for validation and analysis purposes.
- Those people who are going to fill out the forms should have a chance to help design them.
- Checklists are convenient for both collecting and analyzing data.

A prototype form was designed to collect the data needed to answer the questions described in the preceding section and circulated to all members of the A-7 project [15]. The form was modified and the process repeated until all were satisfied with the proposed form. It was then briefly tested, and a few minor modifications made.

Data Collection Procedures

Change data collection was made part of the configuration management process for A-7 documents. As documents are completed, they are placed under configuration control, and all changes made to them are described and monitored. Change report forms tailored to the objectives and format of the documents under control are used. Figure 1 is the change report form used for the requirements document.

Proposed changes to baselined documents are submitted on a change report form (CRF). The proposed change is reviewed by the configuration control board (CCB). If disapproved, the change may be returned to the proposer with an explanation. If approved, an A-7 project member is assigned to implement the change. The implemented change is reviewed again by the CCB for correctness. Often, the proposer is a member of the CCB. Also, the proposer is sometimes the same as the implementer.

Integration of change data collection with configuration control has the advantage that no change data is lost as long as the configuration control process works. Furthermore, only one form is needed for both configuration control and data collection. Change-data analysts are thus assured of the completeness of their data. In addition, the proposer and implementer of the change are both identifiable if further information is needed.

A characteristic of the change process is that trails to and from the document and the CRF are maintained. Changed sections are marked both with a symbol to denote that a change has been made and with the number of the CRF describing the change. The CRF always contains the (sub)section(s) changed, and often the page numbers. The change data analyst can easily find the exact part(s) of the document changed.

Data Validation and Analysis

Several times a year the accumulated change report forms are reviewed and an analysis conducted for evaluation purposes. As part of this process, the forms are validated. Experience with previous change data collection projects has convinced us that validation of the forms is essential. Validation includes examination of each form for completeness and consistency. When necessary, interviews with the proposer and the implementer of the change are conducted to obtain missing data and correct errors.

The various kinds of cross-referencing used facilitate both change to the documentation and change data validation and analysis of the kind described in this paper. As an example, during change validation several incompletely implemented changes have been discovered and reported back to the configuration control board.

This report contains the results of the first major evaluation of the changes to the requirements document. These changes cover the period from 19 December 1978 to 15 February 1980 (the document was baselined in November 1978).

RESULTS OF THE DATA ANALYSIS

The answers to the questions posed in the preceding section are presented here, based on data collected during the first 14 months of use of the requirements document.

NRL REPORT 8445

A-7 Requirements Document Change Report Form

Number _____

Current Date.....
Change Started On..

Month	Day	Year

Change Discovery

1. How was the document being used when the need for the change was discovered?

- | | |
|---|---|
| <input type="checkbox"/> Validation review by the authors | <input type="checkbox"/> As a software design reference |
| <input type="checkbox"/> Validation review by non-authors | <input type="checkbox"/> As a coding reference |
| <input type="checkbox"/> As a maintenance reference | <input type="checkbox"/> As a test reference |
| | <input type="checkbox"/> Other: _____ |

Identification

2. Description of change: _____

3. Section(s) Changed Section(s) Examined But Not Changed

Name	Section	Subsection(s)	Subsection(s)
Intro.	0.		
TC 2	1.		
Data Items	2.		
Modes	3.		
Functions	4.		
Timing	5.		
Accuracy	6.		
UE's	7.		
Subsets	8.		
Changes	9.		
Glossary	10.		
Sources	11.		
Data Item Index			
Mode Index			
Function Index			
Dictionary			

Type Of Change

4. Why is the change being made (check one)?

- | | |
|--|---------------------------|
| <input type="checkbox"/> To correct an original error | |
| <input type="checkbox"/> To complete or correct a previous change (Previous CRF # _____) | |
| <input type="checkbox"/> To comply with unexpected requirement change (violates sect. 9 assumption) | |
| <input type="checkbox"/> To comply with expected requirement change (assumed in sect. 9 subsect. __) | |
| <input type="checkbox"/> To remove unnecessary information | |
| <input type="checkbox"/> To reorganize within one section | } Readability Improvement |
| <input type="checkbox"/> To reorganize among several sections | |
| <input type="checkbox"/> Other: _____ | |

5. What was the effort in person-time required to understand and make the change?

!.....!.....!.....!.....!
0 1 man-hr 1 man-day 1 man-week 1 man-month

6. Estimate the percentage of the effort just to understand the change.

!.....!.....!.....!.....!.....!
0 20 40 60 80 100

Fig. 1a - A-7 requirements document change report form (obverse)

BASILI AND WEISS

FOR ERROR CORRECTIONS ONLY

Type Of Error

7. How is the error best characterized (check one)?

- | | | |
|--|---|----------------------|
| <input type="checkbox"/> Clerical | <input type="checkbox"/> Incorrect Fact | } Inappropriate Fact |
| <input type="checkbox"/> Ambiguity | <input type="checkbox"/> Information put in wrong section | |
| <input type="checkbox"/> Omission | <input type="checkbox"/> Implementation fact included | |
| <input type="checkbox"/> Inconsistency | <input type="checkbox"/> Other: _____ | |

For Section 2 (Data Item) Errors

8. Which part(s) of the subsection were incorrect (check all that apply)?

2

- | | |
|--|---|
| <input type="checkbox"/> Hardware | <input type="checkbox"/> Instruction Sequence |
| <input type="checkbox"/> Description | <input type="checkbox"/> Data Representation |
| <input type="checkbox"/> Value Characteristics | <input type="checkbox"/> Comment |

For Section 3 (Modes) and Section 4 (Functions) Errors

How is the error best characterized (check all that apply)?

3

9. Section 3

- ☐ Mode Transition Error
☐ Mode Condition Error
☐ Mode Overview Error

10. Section 4

- ☐ Applicable Mode List Error
☐ Output Data Item Error
☐ Output Description Error
☐ Event Table Error
☐ Selector Table Error
☐ Condition Table Error
☐ Inconsistent or incomplete table

4

11. ☐ Corresponding errors were made in both sections 3 and 4.

For Errors Involving Dictionary Items

12. How is the error best characterized (check all that apply)?

DICT.

- ☐ Incorrect item
☐ Incomplete item
☐ Dictionary inconsistent with usage elsewhere

For Notational Errors

13. Which notation type was in error?

NOT.

- | | |
|-------------------------------------|---|
| <input type="checkbox"/> /input/ | <input type="checkbox"/> operator |
| <input type="checkbox"/> //output// | <input type="checkbox"/> event |
| <input type="checkbox"/> \$value\$ | <input type="checkbox"/> simple condition |
| <input type="checkbox"/> !item! | <input type="checkbox"/> compound condition |
| <input type="checkbox"/> *mode* | |

14. How is the error best characterized?

- ☐ incorrect grouping
☐ incorrect symbol
☐ incorrect value

Comments

15. Please give any information that may help understand the change and its cause.

Name _____

Date _____

Fig. 1b - A-7 requirements document change report form (reverse)

Changes discussed in this report fall into one of two categories: error corrections and non-error-corrections. For the sake of brevity, the term error is used in place of error correction, and the term modification is used in place of non-error-correction. The term changes is used to refer to both error corrections and non-error-corrections where no distinction between the two need be made.

The data distributions presented are generally displayed in accordance with the categories used on the CRF. As an example, error distributions use the categories from part 7 of the CRF: "Clerical," "Ambiguity," "Omission," "Inconsistency," "Incorrect fact," "Information put in wrong section," "Implementation fact included," and "Other."

Before proceeding to an analysis of each of the questions previously listed, we present some of the general characteristics of the data collected. Figure 2 is a distribution of changes by type. Of the 88 changes reported, 79 were errors. Of the 79 errors, 18 were clerical. Figure 3 is a distribution of errors by type. Figures 2 and 3 also show the effort involved in making changes and in correcting errors. Sections of the histograms marked T (denoting trivial) indicate changes that took a man-hour or less of effort to make, those marked E (denoting easy) took more than a man-hour but no more than a man-day, those marked M (denoting moderate) took more than a man-day but no more than a man-week, and those marked F (denoting formidable) took more than a man-month. There were no changes that took more than a man-week but no more than a man-month. Only one formidable error has yet been found.

Data on the effort required to understand and make changes are provided in parts 5 and 6 of the CRF. These data are the basis for our effort estimates. The data supplied do not include secretarial and editing effort, but only that effort required to understand why a change has to be made and what change has to be made and to describe the change in form sufficient for an editor or typist to incorporate it into the document. In addition, nearly all changes were one-person changes; i.e., one person noticed the need for the change, did the research necessary to understand what change had to be made, and proposed the change. Nearly all estimates of the effort to make changes can then be viewed as the effort required of one person. Effort estimates given in this report are obtained by assuming that trivial changes took one-half hour of effort, easy changes one-half day, and moderate changes one-half week.

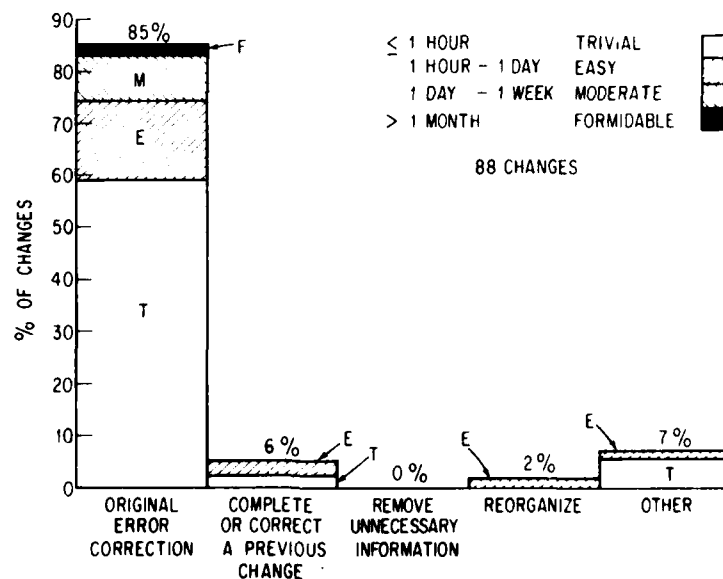


Fig. 2 — Types of changes

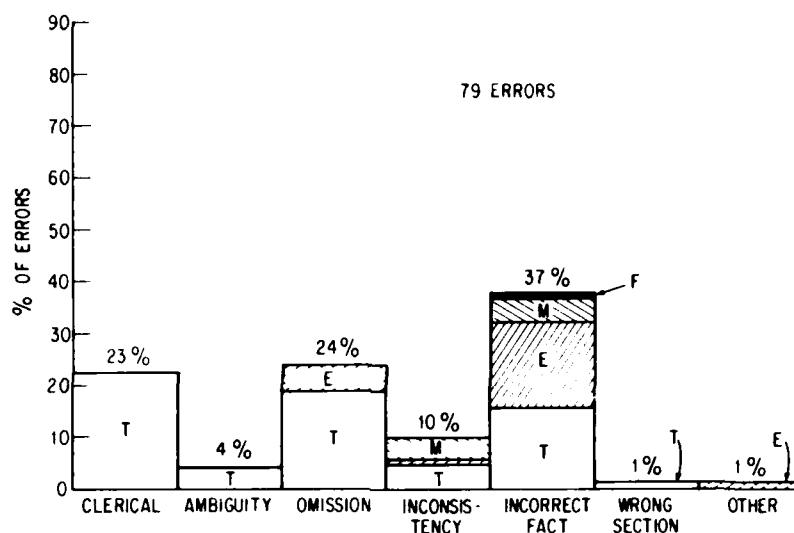


Fig. 3 — Types of errors

An estimate of six man-weeks for the one formidable error was obtained through discussions with the proposer and implementer of the change. It is interesting to note that most of this effort (about 80%) involved understanding what the correction should be.

The effort expended in producing the requirements document originally was 17 man-months, including both development and review. The effort expended in making changes was about 11 man-weeks and the effort in correcting errors was about 10.5 man-weeks. We feel these are small in comparison to the original effort. Discussions with those who wrote and those who changed the document revealed that many of the people who were making changes were not among the original authors; the effort to make the changes consequently contains some learning effort also.

We believe that one reason the requirements document is well maintained is the ease of making individual changes to it. As will be shown in the discussion of question 4, typical changes take about 2.4 hours. These hours are often expended over a relatively long period of time, since the need for a change may be noted without specifying the change to be made.

The list of questions below is separated into three categories: those questions for which we believe there is sufficient data to discern patterns in the data distributions (questions with preliminary answers), those questions for which there is insufficient data, and those questions for which lack of data may be meaningful.

Questions with Preliminary Answers

Are the External Interfaces Specified Correctly?

External interfaces are described in section 2 of the requirements. To answer this question one must find all errors involving section 2. It is also of interest to segregate these errors by type and to estimate the effort involved in correcting them. Figure 4 shows the distribution of nonclerical external-interface errors by type. Clerical errors have been omitted because we assume that the reason for their occurrence is unrelated to the contents of the section of the document in which they occur.

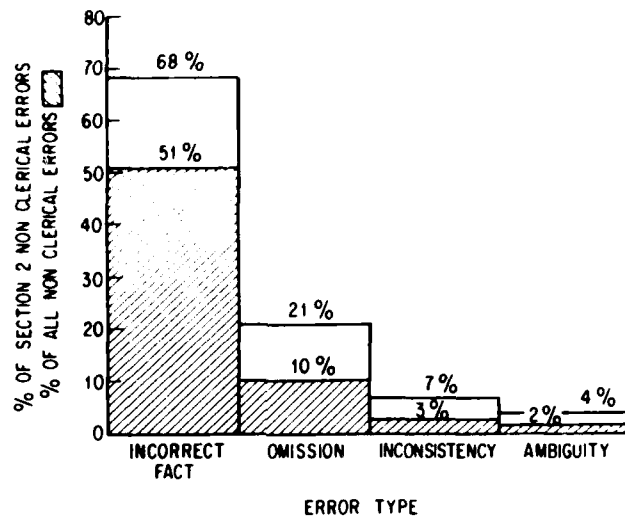


Fig. 4 — Nonclerical errors in section 2

Section 2 of the requirements is of particular interest because it contained the one formidable error found so far. This error involved the incorrect definition of a coordinate system. Most of the effort in correcting it was consumed by a study of the use of coordinate systems, the transformations between them, and the sensors providing navigational information for the aircraft. The effort required to correct this error was greater than the effort required to make all other changes to the document combined.

We estimate the effort to correct the nonclerical errors in section 2 of the requirements as 315 man-hours or about 8 man-weeks. This was far more effort than any other section of the document and about 75% of the total effort to correct nonclerical errors so far. One reason for this may be that section 2 has probably received more use as a design specification than any other section at this stage of the project; consequently, it has received more attention than any other section. This issue will not be settled until the project has ended.

Is the Document Easy To Change?

Part 5 of the CRF provides an estimate of the effort to make the change. Using the previously described effort categories and estimation algorithm, based on the responses to part 5 of the CRF we can estimate the effort needed to make changes of various types to the document. The total effort required for all changes estimated in this way is 442 man-hours, or about 11 man-weeks (note that without the one formidable-class error, the effort would be 202 man-hours, or about 5 man-weeks). The average effort to make a change was 5 man-hours, and the average to correct an error of any type was slightly higher, 5.4 man-hours. Without the formidable error, these figures are sharply reduced, becoming 2.3 and 2.4 man-hours respectively.

Although there are few data on modifications (only nine have yet been reported), the initial indication is that they require less effort than errors, averaging 1.8 man-hours.

Is It Clear Where a Change Has To Be Made?

Because of the skewness of the effort distribution, i.e., nearly 70% of the changes are in the trivial category as shown in Fig. 5, one might consider the "typical" change as requiring 2.4 man-hours.

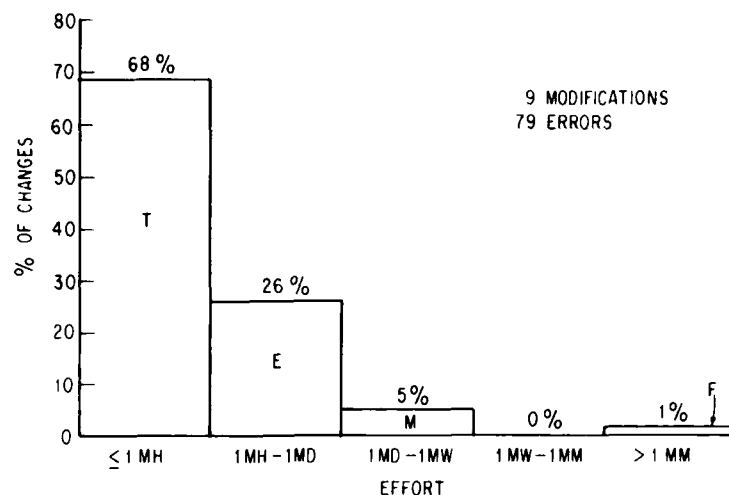


Fig. 5 — Effort to change

Analysis of the data from part 3 of the CRF shows that, for all but one change, only one section of the document had to be examined to make the change. We can now characterize the typical change as taking 2.4 hours and only requiring inspection of one section of the document.

Are Changes Confined to a Single Section?

Figure 6, obtained from the response to part 2 of the CRF, shows the distribution of changes by the number of sections of the document changed. Most changes only required modification of one section of the document. Analysis of the effort for single-section changes compared to multisection changes shows that on the average the latter required about 27% more effort: 4.8 man-hours for single and 6.1 man-hours for multiple. Not only does it seem that the document meets the goal of its authors in this respect, but it also seems that this was a worthwhile goal.

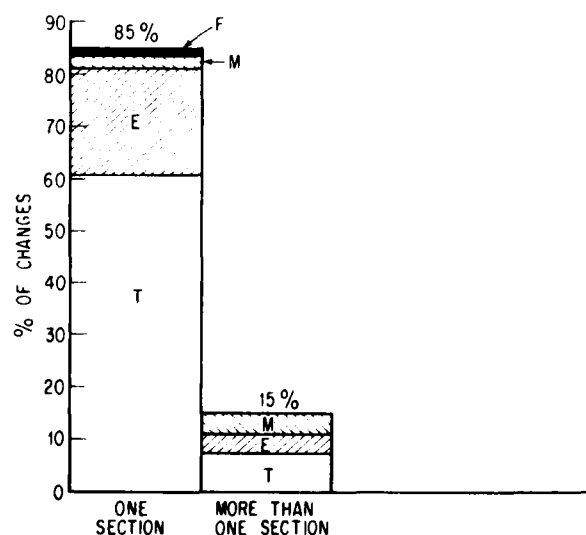


Fig. 6 — Confinement of changes

Which Sections Have the Most Errors?

Figure 7 shows the distribution of nonclerical errors by section. Sections 2 and 4 clearly have the majority of reported errors. This is likely because section 2 has received the most use and section 4 second most at this stage in the development cycle. Further analysis of the data shows that the distributions of error types in these two sections differ. Figure 8 shows these distributions, which must be considered partial as yet.

Which Type of Table Has the Most Errors?

Tables tailored to the A-7 flight software are used liberally throughout sections 2, 3, 4, and the dictionary. The four principal kinds of tables used are mode-transition tables, event tables, selector tables, and condition tables (see [14] for definitions of the different types of tables). Of the 61 nonclerical errors so far discovered, 24, or 39%, were errors involving tables. More than half of these (54%) were found in event tables. The difference in error incidences may possibly be attributed to the difference in nature of the tables. It was possible to do consistency and completeness checks for condition, selector, and mode-transition tables (a procedure for validating condition tables is described in [13]), but not for event tables.

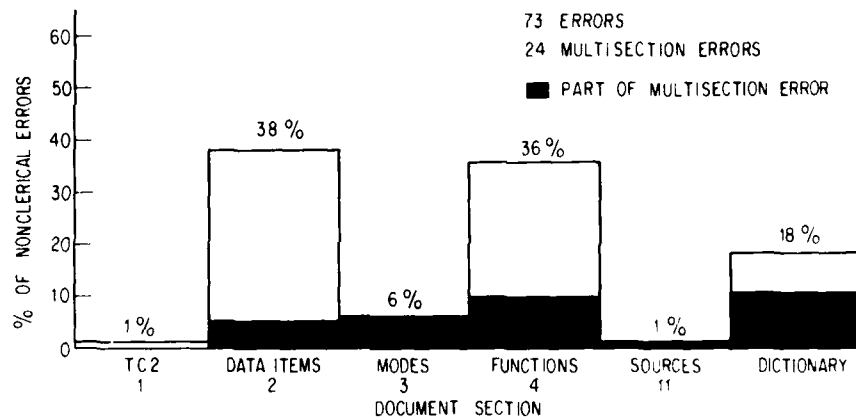


Fig. 7 — Nonclerical errors by section

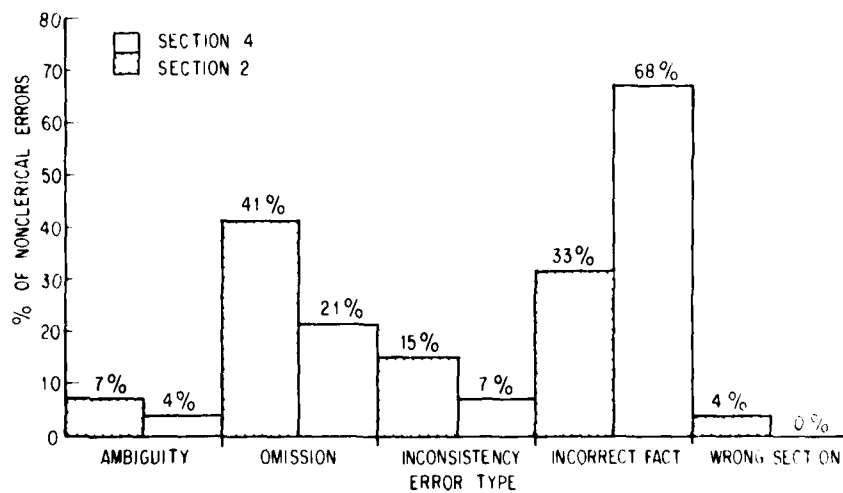


Fig. 8 — Nonclerical errors in sections 2 and 4

The distribution of nonclerical table errors by type of error is shown in Fig. 9. This distribution differs markedly from the corresponding distribution for all nonclerical errors as shown in Fig. 10. Omissions dominate the table errors, whereas incorrect facts dominate the distribution of all nonclerical errors. Furthermore, the margin of domination is smaller for the table errors, and the distribution over omissions, inconsistencies, and incorrect facts is more uniform for them. There are several possible explanations. One is that there are insufficient data yet for the complete pattern to appear. A second is that the tables may be just a good way of organizing information so as to make completeness checks easy.

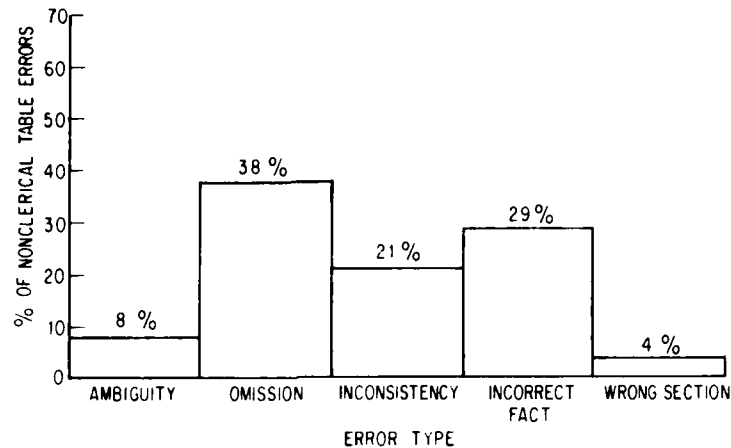


Fig. 9 — Nonclerical table errors

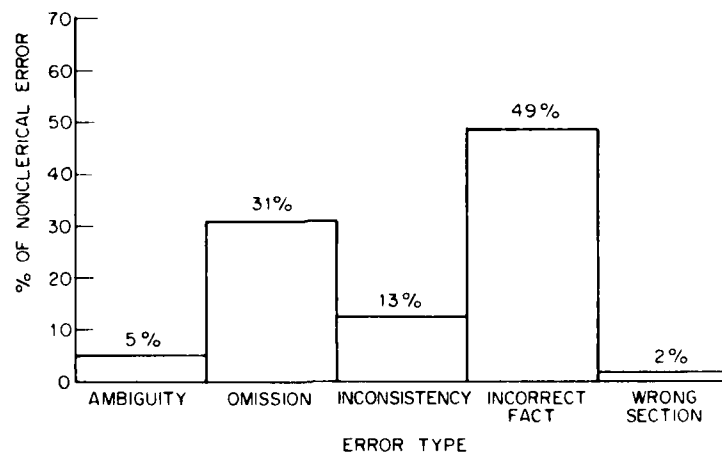


Fig. 10 — Nonclerical errors by type

Because of the relatively small number of nonclerical errors involving tables so far found, it is premature to draw firm conclusions concerning the usefulness of tables in general or event tables specifically from the data. Now that patterns concerning table errors in the partial data have been noticed, we will continue to look for them during the remainder of the development cycle.

Patterns of the sort described in the foregoing provide useful feedback to the developers. Unequal error distributions may mean that some sections of the document have not been as carefully examined or as fully used as others, and require further review.

What Use of the Document Reveals the Most Errors?

Figure 11 shows the distribution of changes according to the way the document was being used when it was discovered that a change had to be made. The distribution is derived from data from part 1 of the CRF. Since the project is currently in the design phase, it is not surprising that most errors have been discovered as a result of using the document as a software design reference. Recall that data collection started after the document was baselined and had already undergone validation. Clearly, a number of errors remained even after the initial validation process was completed. Some of these, but not the majority, were uncovered by later validation reviews. (This applies especially to the mode-transition tables, for which a good validation algorithm was not discovered until after baselining.) The initial validation process included reviews both by the authors and the maintainers of the current A-7 flight software at NWC.

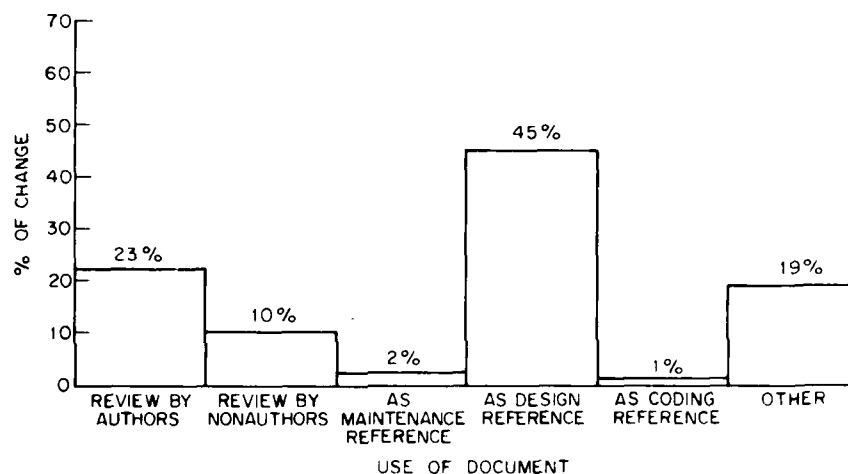


Fig. 11 — Discovery of need for change

Are Sections 3 (Modes) and 4 (Functions) Consistent with Each Other?

Sections 3 and 4 of the requirements document are complementary views of the system. Section 3 describes the set of modes in which the system may operate and the events that cause transitions between modes; it may be viewed as a state description of the system. It also contains information about operations to be performed in the different modes and data items that are used and that may be affected when the system is in the mode. Section 4 describes the functions the system must perform, in which modes it must perform them, and, for each function, the output data items affected by the function.

Because sections 3 and 4 offer complementary views of the system, they should be consistent with each other. An error discovered in section 4 should result in a cross-check to section 3 for a corresponding error. As yet, there have been only two cases where corresponding errors have been found in both sections.

Is The Dictionary Complete, Correct, and Consistent with the Rest of the Document, and Will It Remain So?

The dictionary serves as a convenient and useful means for abbreviating and cross-referencing the requirements document. Terms need only be defined in one place, and those unfamiliar with the

meaning of a particular term can quickly find its definition. The dictionary also serves as an important tool for abstraction. The definition for a term such as slant range may be used without the need to know how to calculate it or what data are needed to calculate it.

Figure 12 shows that most dictionary changes involved the dictionary and at least one other section. In every case but one, these changes were error corrections that included adding a new term to the dictionary. One may think of these changes as adding abstraction to the document. The one exception was a definition that was incomplete.

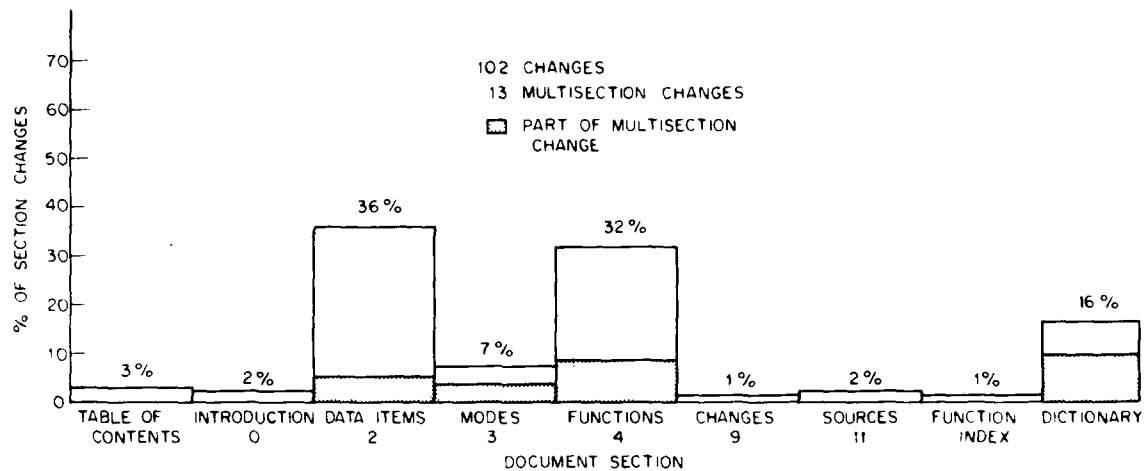


Fig. 12 — Changes by section

Changes that were confined to the dictionary alone were usually incorrect definitions; i.e., the term defined corresponded to some "ideal" quantity whose accepted definition did not quite correspond to the requirements dictionary definition. In only one case was the dictionary found to be inconsistent with usage elsewhere in the document.

For its first 14 months of use, the dictionary appears to have been well maintained. Changes elsewhere in the document stimulated the appropriate changes in the dictionary. There seem to be few inconsistencies with the rest of the document.

Questions for Which Lack of Data May Be Meaningful

Is Externally Visible Behavior Only Specified Without Implying a Particular Implementation?

Data to answer this question are supplied by part 7 of the CRF. Including an implementation fact in the requirements is considered an error. No errors of this type have yet been reported.

Are the Appropriate External Interfaces Specified?

Data to answer this question are supplied by parts 2, 3, and 7 of the CRF. Currently, no change has involved adding a new external interface or deleting an existing interface from the document.

Does the Document Contain Unnecessary Information?

No changes involving removal of unnecessary information (see part 4 of the CRF) have yet been made, nor have any errors involved the inclusion of an implementation fact.

Questions Not Currently Answerable

There are insufficient data available to answer the following questions.

Are the Changes Likely to Occur Predicted Correctly?

Requirements for the NRL version of the A-7 flight program were frozen at the start of the redevelopment project. Consequently, data to answer this question will not become available until the NRL flight program is completed and changes to the program are then considered.

Is the Proper Set of Undesired Events Described?

Is the Notation Used Unambiguous?

Where Do the Most Changes Have To Be Made?

Which Subsections of Sections 2 (Data Items), 3 (Modes), and 4 (Functions) Are Most Error-Prone?

CONCLUSIONS

We have two main objectives in monitoring the changes made to the A-7 software requirements document. One objective is to investigate the feasibility of applying goal-directed data collection concurrently with document maintenance. Similar techniques have been successfully applied to code during program development [17]. A second objective is to try to measure the success with which the A-7 requirements authors met their objectives. We believe the latter objective to be particularly important because the A-7 redevelopers are attempting to use a methodology to produce an engineering model. If they succeed, it is important to know the weak and strong points of the model. If they fail, it is important to know what the troublesome areas are both in the application of particular techniques and in the integration of different techniques.

Two kinds of conclusions may be drawn from this study; one kind concerns the data-collection method itself, and one kind concerns the A-7 software requirements document. Conclusions concerning the data collection method are:

- The data collection method seems to be feasible and useful. By integrating it with the configuration control process we have tried to keep down the overhead associated with it. Data distributions to answer questions of interest both to the A-7 redevelopers and to software engineers are producible.
- Data distributions based on partial data seem to provide some useful feedback to the redevelopers. As an example, error distributions that show uneven patterns of error detection may indicate that some document sections need further attention.
- As patterns are discerned in the data, new questions of interest emerge. As an example, comparison of error distributions across different sections of the document shows that the distributions often differ significantly. There is no obvious explanation for these differences.

but many hypotheses can be formed to explain them. We expect that answers to some of the newly formed questions will be available later in the project; others will probably not be answerable with the data currently being collected.

Conclusions concerning the requirements document are generally answers to the questions discussed in the previous section. Some of the more significant conclusions are:

- The document seems to be easily maintained. The low effort to correct a "typical" error supports this conclusion. It is important to note that all the requirements errors must be found in order to produce a correct system whether or not the requirements document is updated to reflect the corrections. As a result, the effort involved in understanding requirements errors comes free (from the viewpoint of updating the requirements document).
- The document is worth maintaining. The uses to which it is being put, as taken from part 1 of the CRF, show that it is being heavily used during design and is being used for maintenance of the existing A-7 flight software.
- Despite a validation process that included both the authors of the document and the maintainers of the existing flight software, a number of errors remained in the document after it was validated and baselined. The uneven distribution of errors by sections suggests that a significant number of errors may remain in the sections that have been lightly used.
- The document seems to be well structured in that changes can be made in one section without requiring many changes elsewhere.

We expect to continue data collection through the entire A-7 flight software redevelopment project. Data collection will be tailored to the project phase and the techniques being used. We have presented here a description of the data collection techniques and results from analysis of partial data because we would like to encourage others to pursue similar projects.

ACKNOWLEDGMENTS

The authors especially thank all of those who have filled out and are continuing patiently to fill out A-7 change report forms, especially Kathryn Heninger and Alan Parker.

We thank Kathryn Heninger, Dr. Rudy Krutar, Alan Parker, Dr. David Parnas, and Dr. John Shore for their suggestions contributing to the design of the requirements document change report form.

Conversations with Kathryn Heninger and Alan Parker were particularly helpful in analyzing the changes.

Finally, Kathryn Heninger, Dr. Carl Landwehr, and Greg Lloyd provided many comments helpful in improving an earlier draft of this paper.

REFERENCES

1. D. L. Parnas, "Information Distribution Aspects of Design Methodology," in *Information Processing 71: Proceedings of IFIP Congress 71*, North-Holland, Amsterdam, 1972, Vol. 1, pp. 339-344.
2. D. L. Parnas, "On the Criteria to be Used in Decomposing Systems into Modules," *Commun. ACM* 15, 1053-1058 (Dec. 1972).

3. J. Guttag, "Abstract Data Types and the Development of Data Structures," *Commun. ACM* **20**, 396-404 (June 1977).
4. B. H. Liskov and S. N. Zilles, "Specification Techniques for Data Abstractions," *IEEE Trans. Software Eng.* **SE-1**, 7-19 (Mar. 1975).
5. D. L. Parnas and G. Handzel, "More on Specification Techniques for Software Modules," Fachbereich Informatik, Technische Hochschule Darmstadt, W. Germany, 1975.
6. D. L. Parnas, "The Use of Precise Specifications in the Development of Software," in *Information Processing 77: Proceedings of IFIP Congress 77*, North-Holland, Amsterdam, 1977, pp. 861-867.
7. D. L. Parnas, "Use of Abstract Interfaces in the Development of Software for Embedded Computer Systems," *NRL Report 8047*, June 3, 1977.
8. E. W. Dijkstra, "Co-Operating Sequential Processes," in *Programming Languages*, F. Genuys, editor, Academic Press, New York, 1968, pp. 43-112.
9. D. L. Parnas and K. Heninger, "Implementing Processes in HAS," in *Software Engineering Principles* (course notes), Document HAS. 9, Naval Research Laboratory, Washington, D.C., 1978.
10. P. Brinch Hansen, *Operating System Principles*, Prentice-Hall, Englewood Cliffs, N. J., 1973.
11. C. A. R. Hoare, "Monitors: An Operating System Structuring Concept," *Commun. ACM* **17**, 549-557 (Oct. 1974).
12. J. H. Howard, "Proving Monitors," *Commun. ACM* **19**, 273-279 (May 1976).
13. K. L. Heninger, "Specifying Software Requirements for Complex Systems: New Techniques and Their Application," *IEEE Trans. Software Eng.* **SE-6**, 2-13 (Jan. 1980).
14. K. L. Heninger, J. W. Kallander, J. E. Shore, and D. L. Parnas, "Software Requirements for the A-7E Aircraft," *NRL Memorandum Report 3876*, Nov. 27, 1978.
15. D. M. Weiss, "Design of the A-7 Requirements Document Change Report Form," *NRL Technical Memorandum 7503-320*, Dec. 1978.
16. V. Basili, M. Zelkowitz, F. McGarry, et. al., "The Software Engineering Laboratory," University of Maryland Technical Report TR-535, May 1977.
17. D. M. Weiss, "Evaluating Software Development by Error Analysis: The Data from the Architecture Research Facility," *J. Syst. Software* **1**, 57-70 (1979).

DATE
FILMED
-8